We claim:

1.   1.   A machine-readable medium that provides instructions, which when executed by

2.   a set of processors of one or more processors, cause said set of processors to perform

3.   operations comprising:

4.   storing an operation; and

5.   executing the operation idempotently with a network resource process.

1.   2.   The machine-readable medium of claim 1 wherein storing the operations

2.   comprises:

3.   storing the operation in a log as a record;

4.   receiving a commit command; and

5.   moving the record into an atomic database.

1.   3.   The machine-readable medium method of claim 1 wherein storing the operation

2.   comprises:

3.   receiving the operation;

4.   performing lock contention handling for the operation;

5.   storing the operation if a lock contention is not detected; and

6.   generating a lock contention notification if the lock contention is detected for the

7.   operation.

1.   4.   The machine-readable medium of claim 1 wherein the operation is one of a

2.   sequence of operations comprising an atomic transaction.

1.   5.   The machine-readable medium of claim 1 further comprising:

2.   receiving the operation from a first user; and

3   receiving a second operation from a second user.

1   6.   The machine-readable medium of claim 1 wherein the operation is received from

2   a first user concurrently with a second operation received from a second user.

1   7.   A machine-readable medium that provides instructions, which when executed by

2   a set of processors of one or more processors, cause said set of processors to perform

3   operations comprising:

4   storing a sequence of operations; and

5   performing the sequence of operations as an atomic transaction.

1   8.   The machine-readable medium of claim 7 wherein each of the sequence of

2   operations is performed idempotently.

1   9.   The machine-readable medium of claim 7 wherein storing the sequence of

2   operations comprises:

3   performing lock contention handling for each of the sequence of operations;

4   storing the sequence of operations if a lock contention is not detected; and

5   generating a lock contention notification if the lock contention is detected.

1   10.   The machine-readable medium of claim 7 wherein a sequence of operations is

2   received from a first user and a second sequence of operations is received from a second

3   user.

1   11.   The machine-readable medium of claim 7 wherein the sequence of operations is

2   received from a first user in concurrence with receiving a second sequence of operations

3   from a second user.

1   12.   A machine-readable medium that provides instructions, which when executed by

2   a set of processors of one or more processors, cause said set of processors to perform

3   operations comprising:

4   storing an operation in an atomic database; and

5   performing the operation with a network resource process in response to an

6   commit command.

1   13.   The machine-readable medium of claim 12 wherein the operation is performed

2   idempotently.

1   14.   The machine-readable medium of claim 12 wherein the operation is one of a

2   sequence of operations comprising a transaction.

1   15.   The machine-readable medium of claim 12 wherein storing the operation in the

2   atomic database comprises:

3   performing lock contention handling for the operation;

4   storing the operation in the atomic database if a lock contention is not detected;

5   and

6   generating a notification of lock contention if the lock contention is detected.

1   16.   The machine-readable medium of claim 12 wherein the operation is received from

2   a first user and a second operation is received from a second user.

1   17.   The machine-readable medium of claim 12 wherein the operation is received from

2   a first user in concurrence with a second operation received from a second user.

1   18.   A machine-readable medium that provides instructions, which when executed by

2   a set of processors of one or more processors, cause said set of processors to perform

3   operations comprising:

4   receiving an operation;

5   determining if a lock contention exists for a record corresponding to the

6   operation; and

7   generating a notification of the lock contention if a lock contention does exist for

8   the record.

1   19.   The machine-readable medium of claim 18 wherein the operation is one of a

2   sequence of operations comprising an atomic transaction.

1   20.   The machine-readable medium of claim 18 wherein the operation is performed

2   idempotently with a network resource process.

1   21.   The machine-readable medium of claim 18 wherein the operation is received from

2   a first user and a second operation is received from a second user.

1   22.   The machine-readable medium of claim 18 wherein the operation is received from

2   a first user in concurrence with a second operation received from a second user.

1   23.   The machine-readable medium of claim 18 further comprising storing the

2   operation if the lock contention does not exist.

1   24.   A machine-readable medium that provides instructions, which when executed by

2   a set of processors of one or more processors, cause said set of processors to perform

3   operations comprising:

4   storing a first and second operation as an atomic transaction;

5   performing the first and second operation idempotently with a set of network

6   resource processes.

1   25.   The machine-readable medium of claim 24 wherein storing the first and second

2   operation as an atomic transaction comprises:

3   performing lock contention handling for the first and second operation;

4   storing the first and second operation of the atomic transaction in a log if a lock

5   contention is not detected; and

6   generating a lock contention notification if the lock contention is detected.

1   26.   The machine-readable medium of claim 24 wherein storing the first and second

2   operation as the atomic transaction comprises:

3   Storing the first and second operation of the atomic transaction in a log;

4   receiving a commit command for the atomic transaction;

5   indicating the atomic transaction as committed; and

6   storing the atomic transaction in an atomic database.

1   27.   The machine-readable medium of claim 24 wherein the first and second operation

2   are received from a first user and a third operation of a second transaction is received

3   from a second user.

1   28.   The machine-readable medium of claim 24 wherein the second operation is

2   received from a first user concurrently with a third operation received from a second user.

1   29.   A network element comprising:

2   a processor to execute a set of atomic transactions; and

3   a storage unit coupled to the processor, the storage unit to store the set of atomic

4   transactions.

1   30.   The network element of claim 29 wherein each of the set of atomic transactions is

2   comprised of a set of operations, the processors to execute each of the set of operations

3   idempotently.

1   31.   The network element of claim 29 wherein the processor forms lock contention

2   handling for the set of atomic transactions; and the processor generates the lock

3   contention notification if the lock contention is detected.

1   32.   The network element of claim 29 further comprising a set of interfaces coupled to

2   the processor, each of the set if interfaces to receive at least one of the set of atomic

3   transactions, each of the set of interfaces corresponding to a different user.

1   33.   A network element comprising:

2   an interface to receive a plurality of operations from a user;

3   a configuration manager coupled to the interface, the configuration manager to

4   process the plurality of operations; and

5   an atomic database coupled to the configuration manager, the atomic database to

6   store the plurality of operations as a transaction.

1   34.   The network element of claim 33 wherein the configuration manager processes

2   the plurality of operations as an atomic transaction.

1   35.   The network element of claim 33 further comprising:

2   the atomic database to detect lock contention;

3   the configuration manager to generate a notification of the lock contention

4   detected by the atomic database; and

5   the interface to display a message corresponding to the notification generated by

6   the configuration manager.

1   36.   The network element of claim 33 further comprising a second interface coupled to

2   the configuration manager, the second interface to receive a second plurality of

3   operations from a second user.

1   37.   A computer implemented method comprising:

2   storing an operation; and

3   executing the operation idempotently with a network resource process.

1   38.   A computer implemented method of claim 37 wherein storing the operations

2   comprises:

3   storing the operation in a log as a record;

4   receiving a commit command; and

5   moving the record into an atomic database.

1   39.   The computer implemented method of claim 37 wherein storing the operation

2   comprises:

3   receiving the operation;

4   performing lock contention handling for the operation; storing the operation if a

5   lock contention is not detected; and

6   generating a lock contention notification if the lock contention is detected for the

7   operation.

1   40.   The computer implemented method of claim 37 wherein the operation is one of a

2   sequence of operations comprising an atomic transaction.

1   41.   The computer implemented method of claim 37 further comprising:

2   receiving the operation from a first user; and

3   receiving a second operation from a second user.

1   42.   The computer implemented method of claim 37 wherein the operation is received

2   from a first user concurrently with a second operation received from a second user.

1   43.   A computer implemented method comprising:

2   storing a sequence of operations; and

3   performing the sequence of operations as an atomic transaction.

1   44.   The computer implemented method of claim 43 wherein each of the sequence of

2   operations is performed idempotently.

1   45.   The computer implemented method of claim 43 wherein storing the sequence of

2   operations comprises:

3   performing lock contention handling for each of the sequence of operations;

4   storing the sequence of operations if a lock contention is not detected; and

5   generating a lock contention notification if the lock contention is detected.

1   46.   The computer implemented method of claim 43 wherein a sequence of operations

2   is received from a first user and a second sequence of operations is received from a

3   second user.

1 47. The computer implemented method of claim 43 wherein the sequence of

2 operations is received from a first user in concurrence with receiving a second sequence

3 of operations from a second user.

1 48. A computer implemented method comprising:

2   receiving an operation;

3   determining if a lock contention exists for a record corresponding to the

4   operation; and

5   generating a notification of the lock contention if a lock contention does exist for

6   the record.

1 49. The computer implemented method of claim 48 wherein the operation is one of a

2 sequence of operations comprising an atomic transaction.

1 50. The computer implemented method of claim 48 wherein the operation is

2 performed idempotently with a network resource process.

1 51. The computer implemented method of claim 48 wherein the operation is received

2 from a first user and a second operation is received from a second user.

1 52. The computer implemented method of claim 48 wherein the operation is received

2 from a first user in concurrence with a second operation received from a second user.

1 53. The computer implemented method of claim 48 further comprising storing the

2 operation if the lock contention does not exist.